An Introduction to R for Epidemiologists using RStudio
indexing

Steve Mooney, stealing heavily from C. DiMaggio

Department of Epidemiology
Columbia University
New York, NY 10032
sjm2186@columbia.edu

An Introduction to R for Epidemiologists using RStudio
Indexing in R
SER Summer 2014
Outline

1. Indexing Overview
2. Indexing Vectors
3. Indexing Matrices & Arrays
4. Indexing Lists
5. Indexing Dataframes
Indexing Overview

Why Indexing

Indexing is how you refer to data within a data structure

1. To read out values (e.g. to plot)
2. To clean data
3. To format output
Outline

1. Indexing Overview
2. Indexing Vectors
3. Indexing Matrices & Arrays
4. Indexing Lists
5. Indexing Dataframes
indexing vectors
myVector[n]

people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
people[1]
people[4]
people[6]
people[-1]
people[c(2,4)]
people[c(4,2)]
Indexing Vectors

**sorting vectors**

**sort() rearranges the same vector**

```r
x <- c(12, 3, 14, 3, 5, 1)
sort(x)
rev(sort(x))
```

**sort() does not change the vector**

```r
sort(x)
x
x <- sort(x)
x
```
ordering and ranking vectors

You often want to sort one vector by values in another

**order() to rearrange another vector**

```r
ages<- c(8, 6, 7, 4, 4)
order(ages)
people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
people[order(ages)]
```

creates an *index* of positional integers to rearrange elements of *another* vector, e.g. `people[c(4,5,2,3,1)]`, 4th element (Danielle) in 1st position, 5th element (Eunice) in 2nd position, 2nd element (Bob) in 3rd position, etc...

**rank() doesn’t sort**

```r
x <- c(12, 3, 14, 3, 5, 1)
rank(x)
```
Using indexing for data cleaning

myVector[n] <- new value

```r
people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
people
people[2] <- "Robert"
people
```
Modification using complex indices

```r
people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
people
people
people[-2] <- c("Alison", "Charles", "David", "Eleanor")
people
```
Indexing by logical

```r
people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
which.people <- c(TRUE, FALSE, FALSE, TRUE, TRUE, TRUE)
people[which.people]
which.people <- people == "Bob"
which.people
people[which.people]
which.people <- people %in% c("Bob", "Charlie")
which.people
people[which.people] <- "Not to be named"
people
```
missing values

- `is.na()` - returns logical vector of NA positions
  useful for replacing missing values
- `!is.na()` - positions that do not contain NA

```r
x <- c(10, NA, 30)
is.na(x)
x[is.na(x)] <- 999
x
```
other unusual values

- `is.nan()` - not a number
- `is.infinite()` - infinite

```r
x <- c(10, 0, 30)
y <- c(0, 0, 2)
z <- x/y
z
is.infinite(z)
is.nan(z)
```
Indexing by name

```r
people <- c("Alice", "Bob", "Charlie", "Danielle", "Eunice")
names(people) <- c("President", "Vice-President", "Secretary", "Staff", "Intern")
people
people["President"]
people[c("President", "Intern")]
people[c("President", "Intern")]<- "Vacant"
people
```
Indexing Matrices & Arrays

Outline

1. Indexing Overview
2. Indexing Vectors
3. Indexing Matrices & Arrays
4. Indexing Lists
5. Indexing Dataframes
a matrix is a 2-dimensional vector...

...so index each vector

Index a matrix with matrixname[row, column]

```r
myMatrix <- matrix(c("a", "b", "c", "d"), 2, 2)
myMatrix
myMatrix[1, 1]
myMatrix[1, 2]
myMatrix[2, 1]
myMatrix[c(TRUE, FALSE), c(TRUE, FALSE)]
```
Indexing a whole row or column
leave out the row or column

Index a matrix with matrixname[row, column]

```r
myMatrix <- matrix(c("a","b","c","d"),2,2)
myMatrix
myMatrix[1,]
myMatrix[,2]
myMatrix[,2] <- c("e", "f")
```
Indexing an array

Index an array with arrayname[row, column, depth]

```r
ugdp.age <- c(8, 98, 5, 115, 22, 76, 16, 69)
ugdp.age <- array(ugdp.age, c(2, 2, 2))
ugdp.age[1,2,1]
```
Outline

1. Indexing Overview
2. Indexing Vectors
3. Indexing Matrices & Arrays
4. Indexing Lists
5. Indexing Dataframes
a list is a collection of unlike elements

- double brackets \([...]\) index the list items
- `object$name` if a named list

```r
x <- 1:5
y <- matrix(c("a","c","b","d"), 2,2)
z <- c("Peter", "Paul", "Mary")
mm <- list(x, y, z)
mm[[2]]
mm[[2]][2,2]

nn <- list(numbers=x, twoxtwo=y, names=z)
nn$names
nn$names[2]
```
Outline

1. Indexing Overview
2. Indexing Vectors
3. Indexing Matrices & Arrays
4. Indexing Lists
5. Indexing Dataframes
dataframes

2-dimensional tabular lists with equal-length fields
each row is a record or observation
each column is a field or variable (usually numeric vector or factors)

"a list that behaves like a matrix"
dataframes

Option 1: index observations or rows or columns like a matrix

titanic<-read.csv(  
stringsAsFactors=F)  #load titanic data
str(titanic)
titanic[1,2]
titanic[1,2]
head(titanic[,2])
dataframes

Option 2: index columns like a list

titanic$sex
titanic$sex
head(titanic$name)
titanic$sex
mode(titanic$sex)
table(titanic$sex)
table(titanic$sex, titanic$survived)
dataframes
tabular epi data sets

Often index like a matrix to subset rows, then like a list to perform analyses

```r
men <- titanic[titanic$sex=="male",]
table(men$survived)
```